Husky11 Development Board

Getting Started Manual

Version 2.70 for Husky11 Rev. A board

Table of Contents

GETTING STARTED with BUFFALO Monitor 2

GETTING STARTED with Wytec monitor and WBUG11 7

Single Board Computer application 12

ON-BOARD HARDWARE 15

BUFFALO I/O ROUTINES 20

IMPORTANT NOTES 21

The Husky11 board, a low cost and high performance development board, provides real time emulation for the Freescale 68HC11microcontroller A and E families. It offers all useful features of the Freescale EVB board with BUFFALO monitor and adds numerous enhancements at low cost. It combines a complete 68HC11 development system, an advanced trainer, a reliable 68HC711E9, E20 programmer and a versatile Single Board Computer into a single package. For engineers, it serves as a WICE in-circuit emulator development system, a Freescale EVB, EVM and EVS replacement, a convenient prototype platform, and a low cost single board computer. For students, it acts as a user-friendly microcontroller trainer. It is as powerful as a high priced real-time in-circuit emulator, but it is as affordable as a low cost microcontroller evaluation board.

The Husky11 includes easy-to-use and user-friendly IDE software which runs under Windows® 95, 98, 2000, XP and Vista. It offers fast file transfer, single-stepping, breakpoints, data watch for memory and registers, symbolic debugging compatibility with most assemblers and compilers, and user program termination with <Esc> key.

Our exclusive ***phantom monitor***™ technology preserves all interrupt vectors including RESET. The monitor also preserves all on-chip RAM ($00-$1FF), EEPROM, and 31K external emulation RAM ($8400-$FFFF) available for user applications - there is no pre-empted chip memory.

Because many students are taught with the Freescale BUFFALO, we installed the BUFFALO monitor in U2 (27C256 EPROM) The board can be booted from the BUFFALO monitor, so students can use the board immediately before leaning how to use the board with Wytec's monitor.

The hardware includes a prototype area, 16 extra I/O lines in port F and G, one logic probe, on-board 16X2 LCD display with backlight, 4x4 keypad, SPI port, speaker, 4-digit LED display, potentiometer, 8 LED indicators for port B, one 8 position DIP switch connected to port C, 3 pushbutton switches, dual UARTs, RS485 interface, IR transceiver with on-board 38 KHz OSC and 60-pin EVB/EVBU-compatible male / female connectors and 40-pin EVM compatible connector.

The package also includes a 9V 500mA-1A wall plug-in switching power supply and a USB cable.

The specification of the AC adapter is:
DC input: 110V
DC output: 9V
Current rating: 500mA-1A
Type of plug: 2.1mm female barrier plug, center positive

The AC adapter is only available to countries that use 110V.

WARNING: If more power is needed in a robot or other applications, the user should upgrade the AC adapter. Otherwise, the board could keep resetting itself when the VCC drops below 4.6V.
Do not apply a DC voltage higher than 9V to this board.

People often use different terminology. In our product menus, "Download" means to transfer a file from the PC to the EVBplsu2 board, while "Upload" means to transfer a file from Husky11 board to PC.

Through out the manual, **left click** means that you click the left button of the mouse and **right click** means that you click the right button of the mouse.

Install Ep2IDE software:

If you have already installed the Ep2IDE for our WICE emulator or one of our EVBplus boards on your hard drive, you must rename the current folder from c:\Ep2IDE to c:\Ep2IDE_old before installing new software. The installation is automated by running the "**SETUP.BAT**" . It will **OVERWRITE** the current folder if you don't rename it.

After software is successfully installed, you can make a shortcut to AsmIDE.exe.

It's very important to make a shortcut so that its target location is C:\Ep2IDE, not c:\Windows\desktop or other locations. First, right click the Start button. Then, left click "Explorer". Left click on C:\Ep2IDE. Right click on AsmIDE.exe (an application program). Left click "Send to" and finally left click "Desktop" (do not click "COPY"). It will create an icon named "shortcut to AsmIDE" on the desktop. You can double check the target location by right clicking on the icon. Then, left click on "properties". You should see that the target location is C:\Ep2IDE. If you want to make a shortcut for AsmIDE on the Desktop, this is the correct way to do so. If you don't follow this method, you may have a problem running your program. Never drag the AsmIDE.exe to the desktop folder.

The default setting of AsmIDE for the Husky11 board is created in a text file named c:\Ep2IDE\AsmIDE.ini. In the future if you get lost with all the changes, you always can copy this file into the folder named c:\Ep2IDE.

GETTING STARTED with BUFFALO Monitor

The Memory Map with the BUFFALO monitor:

$0000-$00FF	On-chip 256 byte RAM for the 68HC11A1
$0000-$01FF	On-chip 512 byte RAM for the 68HC11E1
$002D-$00FF	On-chip RAM used by BUFFALO monitor
$1000-$103F	On-chip 64 control registers, including port B ($1004)and port C ($1003)
$1003	Port C for DIPswitch
$1004	Port B for LEDs
$2200-$23FF	65C22
$2201	Port F, LCD port
$2203	DDRF
$2200	Port G, Keypad port
$2202	DDRG
$2400-$25FF	68B50
$2400	UCTRL, USTAT
$2401	UART
$2800-$2FFF	External device, /CS on the pin40 of J1
$3000-$7FFF	20K EEPROM of U3
$8000-$DFFF	RAM for user code or data
$E000-$FFFF	BUFFALO monitor firmware in U2

The important default Jumper Settings:

J9	Set for RUN (left side)
J15	Set for BUFL (middle position)
J22	The jumper is installed in the "left" side for enabling single-step function.
J31	The jumper is installed in the "bottom" position for USB interface
J32	Both jumpers are installed in the "top" positions for connecting ACIA (68B50) to USB port.

Any attempt to write to the locations ($2200-$25FF) will have unpredictable results and must be handed with care in user programs.

Because most textbooks are written for BUFFALO monitor, the Husky11 board comes with both BUFFALO monitor and Wytec monitor installed in U2. If you have worked with the BUFFALO monitor in the past, you can use the board right away. In fact, if you use the BUFFALO monitor with this board, the board becomes a standard 68HC11 EVB board just like many other EVBs on the market today, except it offers more on-board peripherals.

Before testing the board with BUFFALO monitor, place a jumper in the middle position (labeled with 'BUFL' which stands for BUFFALO Monitor) of J15 and another jumper on the left position (labeled with 'TRACE') of J22. To test the board, follow the steps 1 through 6 below:

Step 1.

Plug the AC adapter into a wall outlet, and plug the DC plug at the other end into the DC jack on the lower right corner of the Husky11 board. During power up, the RESET LED should flash twice and all other LEDs must be off, the speaker should chirp once (If the chirp is too soft you can remove the sticker on the speaker to increase the volume) and the LCD should display the following message:

"HUSKY11 TRAINER "
" BUFFALO V4.02 "

If this does not occur, turn over to the Questions & Answers section of the Wytec user manual.

Step 2.

Plug the USB cable to the USB jack P2 on the **upper right** corner of the Husky11 board. Plug the other end of the USB cable into a USB port of your PC. Make sure that the jumpers on the J31 and J32 are set correctly for USB interface for the ACIA (68B50). The header J33 is the 68HC11 SCI port in TTL interface that can be used by a user's application program.

Step 3.

Press the reset button on the Husky11 board momentarily and the RESET LED, which is located above the reset button, should flash twice. If it does not flash, turn over to the Questions & Answers section of the Wytec user manual.

Step 4.

To invoke the AsmIDE, right click the Start button, then left click "Explorer" and left click on C:\Ep2IDE and finally, double left click on AsmIDE.exe. If you have created a shortcut icon on the desktop, just double click the icon on the desktop.

Warning note: Always plug the USB cable into Husky11 before invoking the AsmIDE and close the AsmIDE before unplugging the USB cable, Otherwise the AsmIDE may hang up and you need to re-establish the USB link again. The Wytec debugger is a DOS program. When the USB link is broken, you will get an error message such as "NTVDM has encountered a system error". At that time you have to close the Wytec Debugger and AsmIDE, then cycle the USB connection before re-invoke the AsmIDE and Wytec Debugger.

In case the AsmIDE hangs up, you need to close the AsmIDE first, then pull the USB cable out the USB jack P2, wait for a few seconds before re-plug the USB cable into the P2. After cycling USB connection, the PC may re-establish the USB communication. If this does not work, you need to reset your PC. This problem will happen to any IDE, so in order to avoid it, always close the AsmIDE before unplugging the USB cable.

The screen is divided into two windows. The top window is for editing your source code and the bottom window is shared by the **message window** and the **terminal window**.

Step 5:

You only need to use three commands from the AsmIDE for your 68HC11 development work. Use the File command to edit your source code, the Build->Assemble command to assemble your source code, and the Build->Download command to communicate with the Husky11 board.

In the View->Option->Terminal Window Options menu, set the COM port as 1 or 2 to match the COM port number that is assigned to the USB port by Device Manager in control panel. Also, set the COM port options at 9600, N,8,1, and check the "enable the terminal window" box which will disable Wytec hc11 tools.

In the View->Option->Assembler menu, make sure that the chip family is 68HC11 and you can also see that the default assembler name for the hc11 is as11.exe. If you would like to use your own assembler, you can replace the as11.exe.bat with the name of your new assembler.

If the assembler detects an error, it will show the error's line number in the file along with an error message.

If the terminal options are set correctly, the jumper is installed on the BUFL position of J15, and the com port number is correct, you should see the following sign-on message every time the reset button is pressed. If you do not see this, the bottom window may be set for message window. Click the terminal button in the bottom window to enable the terminal window display.

```
HUSKY11: BUFFALO 4.02cc - Bit User Fast Friendly Aid to Logical Operation
```

Press the Enter key, you will get the BUFFALO monitor prompt
>

Step 6

All sample programs are debugged and tested for your convenience. They are located in the folder named c:\Ep2IDE\Ex_BUFFL. Here are the steps to run your first sample program:

1. Click the File button to load test.asm from c:\Ep2IDE\Ex_BUFFL to view this program.

2. Click Build -> Assemble or click the assembler button on the toolbar to assemble your code and generate the test.s19 file. In order to display assembler messages, the bottom window is switched to the message window.

3. Click the terminal button of the bottom window to activate the terminal window and make sure that the BUFFALO monitor prompt '>' is shown on the terminal window. If this is not the case, press the Enter key.

4. At the prompt '>', type "**LOAD**" <Enter>.

5. Click Build->Download and select the file c:\Ep2IDE\Ex_BUFFL\test.s19 to download it.
   Note: If the top window is not loaded with a file, clicking the Build button will prompt you with the "save changes" message. You can ignore that message and click the "No" answer.

6. After the download is done, type G D000 to run the program.

It will run the TEST program in real time. The program will test switches, scan keypad, send a message to the LCD display, adjust LED brightness, generate music. At first you can press the PC0, PC1 or PC2 pushbutton switch and see that the PB0, PB1 or PB2 LED follows the state of the pushbutton switch and then press the PA0 switch to generate sound.

To stop the program, you have to press the reset button momentarily.

All example programs are fully debugged, so the assembler won't generate an error report. If you have an error in your program, you must correct it before an s19 file can be generated.

You can try to run a different example program later after you have finished reading this manual. You should always press the reset button before downloading a new program, because a new program may not work if an interrupt was enabled by a previous program.

Software development with the BUFFALO monitor:

We are using the AsmIDE as a terminal program with the following instructions to create your source code. If you are using a different terminal program, the instructions may vary.

The steps to create your source code are as follows:

1 Click the **File** button to open an existing file or create a new file.

The memory locations from $00-$2Cfor the A family or $00-$2C and $100-$1FF for the E family are available as user DATA RAM. The BUFFALO monitor uses the RAM locations at $2D-$FF. The 23K memory locations from $8400 to $DFFF are available to user program CODE or DATA. In assembly language, you specify the starting address of your CODE by an ORG statement.

You can start DATA RAM at address $00 with the statement ORG 0 followed by RAM variables, as shown below:

```
            ORG   0
TEMP:       RMB   1          ; reserve one byte of RAM for temp storage
XTEMP:      RMB   2          ; reserve two bytes of RAM for temp storage
```

If your program is small, say less than 4K, you can start your program at address $D000 with the statement ORG $D000 followed by your program, as shown below:

```
            ORG   $D000
            LDS   #$8FFF     ; initialize stack point
```

It will assemble your source program and generate hex code within 4K locations from $D000 to $DFFF. If your program is larger, you can change ORG $D000 to ORG $C000, or ORG $A000. You cannot use ORG $E000 or ORG $F000 because the BUFFALO monitor occupies the highest 8K locations ($E000-$FFFF)

With the BUFFALO monitor, the Husky11 board cannot abort your program if it's hung in a loop. The only way to abort it is to reset the board. The problem with resetting the board is that you would not know where the 68HC11 was hung. It also would leave SWI instructions on all breakpoint addresses. You will have to re-download your s19 file all over again.

Here is a very simple program, but it's complete. It will flash the PB7 LED when it's running.

For a good programming practice, you should always place a LDS instruction in the first line of your code.

```
PB7:        EQU   $80              ; bit 7 of port B
FLS_RATE:   EQU   $8B00            ; change this number will change LED flash rate
            ORG   $D000
START:      LDS   #$8FFF
            LDX   #$1000           ; X register points to the register block
BACK:       BSET  4,X PB7          ; turn on the PB7 LED by setting PB7=1
            JSR   DELAY
            BCLR  4,X PB7          ; turn off the reset LED by resetting PB7=0
            JSR   DELAY
            JMP   BACK
DELAY:      LDY   #FLS_RATE
DLY:        DEY
            BNE   DLY
            RTS
            END
```

2. Save your file frequently while editing. If you are creating a new file and giving the file a name to save, enter the file name including the file extension, such as “test.asm”, not just “test”.

3. Click Build button-> Assemble, or click the assembler button on the toolbar to assemble your code and generate an s19 file. If the assembler detects an error, the error message will show the line numbers in your source code that caused errors. Beware that sometimes (not very often, but it does happen) the freeware assembler may indicate a wrong error line in the file and the actual line that caused error may be one line off from the error line number.

WARNING: The free assembler will generate an error on the BSET, BCLR, BRSET and BSCLR instructions if more than one commas are used in those instructions. For instance, BSET 0,X,$20 or BRSET 0,X,$20,$F000 is illegal, but BSET 0,X $20 or BESET 0,X $20 $F000 is OK. You have to use a space character to separate fields, not a comma.

4. Go to the lines with errors and correct them, then go back to the step 3 until there are no errors.

GETTING STARTED with Wytec monitor and WBUG11

The Husky11 board is a standard EVB board with the BUFFALO monitor, but it's an easy-to-use and powerful In-circuit Emulator type of development system when it's under the control of the Wytec Symbolic Debugger, WBUG11. If you don't use the WBUG11, you will not be able to exploit all of the features. Once you have used the WBUG11, you probably would not want to use the BUFFALO again.

The Memory Map with the Wytec monitor:

$0000-$00FF	On-chip 256 byte RAM for the 68HC11A1
$0000-$01FF	On-chip 512 byte RAM for the 68HC11E1
$0800-$0FFF	Wytec monitor firmware in U2
$1800-$1FFF	Wytec monitor firmware in U2
$1000-$103F	On-chip 64 control registers, including port B ($1004)and port C ($1003)
$1003	Port C for DIP switch
$1004	Port B for LEDs
$2200-$23FF	65C22
$2201	Port F, LCD port
$2203	DDRF
$2200	Port G, Keypad port
$2202	DDRG
$2400-$25FF	68B50
$2400	UCTRL, USTAT
$2401	UART
$2800-$2FFF	External device, /CS on the pin40 of J1
$3000-$7FFF	20K EEPROM of U3
$8000-$83FF	RAM used by Wytec monitor
$8400-$FFFF	RAM for user code or data

The important Jumper Settings:

J9 Set for RUN (top position)

J15 Set for Wytec monitor (top position)

J22 The jumper is installed in the "left" side for enabling single-step function in BUFFALO monitor mode. If a user program does not use PA3, leave the jumper in the "left" side. If it does, then move the jumper to the "right" side so the PA3 is not connected to XIRQ. But don't forget to move it back to the "left" side when the BUFFALO monitor is used for debugging, otherwise the single step function in BUFFALO monitor won't work.

Any attempt to write to the locations ($2200-$25FF, $8000-$83FF) will have unpredictable results and must be avoided in user programs.

Before testing the board with the Wytec monitor, move the jumper in the J15 to the "top" position (labeled with 'WYTEC' which stands for Wytec monitor). To test the board, follow the steps 1 through 7 below:

Step 1.

Plug the AC adapter into a wall outlet, and plug the DC plug at the other end into the DC jack on the lower right corner of the Husky11 board. During power up, the RESET LED should flash 4 times and all other LEDs must be off, the speaker should chirp once (If the chirp is too soft you can remove the sticker on the speaker to increase the volume) and the LCD should display the following message:

"HUSKY11 TRAINER "
" WYTEC V4.02 "

If this does not occur, turn over to the Questions & Answers section of the Wytec user manual.

Step 2.

Plug the USB cable to the USB jack P2 on the **upper right** corner of the Husky11 board. Plug the other end of the USB cable into a USB port of your PC. Make sure that the jumpers on the J31 and J32 are set correctly for USB interface for the ACIA (68B50). The header J33 is the 68HC11 SCI port in TTL interface that can be used by a user's application program.

Step 3.

Press the reset button on the Husky11 board momentarily and the RESET LED, which is located above the reset button, should flash twice. If it does not flash, turn over to the Questions & Answers section of the Wytec user manual.

Step 4.

To invoke the AsmIDE, right click the Start button, then left click "Explorer" and left click on C:\Ep2IDE and finally, double left click on AsmIDE.exe. If you have created a shortcut icon on the desktop, just double click the icon on the desktop.

Warning note: Always plug the USB cable into Husky11 before invoking the AsmIDE and close the AsmIDE before unplugging the USB cable, Otherwise the AsmIDE may hang up and you need to re-establish the USB link again. The Wytec debugger is a DOS program. When the USB link is broken, you will get an error message such as "NTVDM has encountered a system error". At that time you have to close the Wytec Debugger and AsmIDE, then cycle the USB connection before re-invoke the AsmIDE and Wytec Debugger.

In case the AsmIDE hangs up, you need to close the AsmIDE first, then pull the USB cable out the USB jack P2, wait for a few seconds before re-plug the USB cable into the P2. After cycling USB connection, the PC may re-establish the USB communication. If this does not work, you need to reset your PC. This problem will happen to any IDE, so in order to avoid it, always close the AsmIDE before unplugging the USB cable.

The screen is divided into two windows. The top window is for editing your source code and the bottom window is shared by the **message window** and the **terminal window**.

Step 5.

To invoke AsmIDE, right click the Start button, then left click " Explorer", then left click on C:\Ep2IDE and finally, double left click on AsmIDE.exe. If you have created a shortcut icon on the desktop, just double click the AsmIDE icon on the desktop.

Step 6:

The default settings of the AsmIDE for the Husky11 board are created in a text file named c:\Ep2IDE\AsmIDE.ini. In the future if you lose with all the changes, you always can copy the original file into the folder named c:\Ep2IDE.

In the View -> Options -> Wytec hc11 tools menu, you must check the 'Wytec Tools Enabled' box to use the Wytec Debugger software, WBUG11. You can change the COM port number, but you do not have to change its baud rate, because the WBUG11 will set it at 38.4K for you. Please note that the default debugger startup batch name is **wbug11.exe**, not mydebug.bat. This is the change we have made from the original AsmIDE.

Step 7:

All sample programs are debugged and tested for your convenience and they are located in the folder named c:\Ep2IDE\Ex_WYTEC. Here are steps to run your first sample program:

1. Click the File button to load test.asm from c:\Ep2IDE\Ex_WYTEC to view the program.

2. Click Build -> Assemble or click the assembler button on the toolbar to assemble your code and generate the test.s19 file.

3. Click Build -> Wytec hc11 tools or click the debugger button on the toolbar that will bring up a small dialog window. You can see the file to be downloaded is c:\Ep2IDE\Ex_WYTEC\test.s19 **(The current download s19 file name is always updated by the assembler).**

4. Now you **MUST** click the "Select this file" button, which will store the file name test.s19 into a text file named c:\Ep2IDE\Ep2IDE.txt. The debugger can automatically download the test.s19 by reading the Ep2IDE.txt when it is invoked. This is a very **important step** and you must do it when you want to debug a different program.

5. Click the Debugger button and the debugger will automatically download the test program TEST.s19. The "Programmer" button is used to program the on-chip EPROM of a 68HC711E9 and you can ignore it for the time being.

6. At the prompt Ep2711>, enter g \start (the label is case sensitive) <Enter>.

It will run the TEST program in real time. The program will test the switches, scan keypad, send a message to the LCD display, emulate an IR proximity sensor, adjust LED display brightness, generate music and shift number 0 to F on seven-segment LED display. At first, you can press the PC0, PC1 or PC2 pushbutton switch and see that the PB0, PB1 or PB2 LED follows the state of the pushbutton switch and then press the PA0 switch to generate sound.

You can press the ESC key on PC keyboard to stop the program. If you stop the program, the speaker may generate clicking noise because the output comparator is still interrupting the 68HC11. To stop the clicking noise, you will need to press the reset button momentarily.

All example programs are fully debugged, so the assembler won't generate an error. If you have an error in your program, you must correct it before an s19 file can be generated.

You can try to run a different example program later after you have finished reading this manual. You should always press the reset button before downloading a new program, because a new program may not work if an interrupt was enabled by a previous program.

Step 8:

To run another program, activate the AsmIDE from the task bar at the bottom of the screen (this will minimize the debugger window). Click Build -> Wytec hc11 tools or click the debugger button on the toolbar that will bring up the Wytec hc11 tools dialog window. Use the browser to choose the s19 file you want to download, such as ex2.s19, then **click the "Select this file"** button and close the dialog window (Do not click the Debugger button, because the debugger was already invoked). Now activate the **WBUG11from the task bar** and press the F10 function key and R option. The ex2.s19 will be automatically downloaded. At the prompt Ep2711>, type "g \start" <Enter>.

Warning: Do not attempt to have more than one debugger window open at the same time. If you do, your PC may lose communication with the board, so you have to close the extra debugger window.

Software development with the Wytec's monitor:

The steps to create your source code as follows:

1 Click the **File** button to open an existing file or create a new file.

The memory locations from $00-$FF for the A family or $00-$1FF for the E family are available as user DATA RAM. The Husky11 board does not use the RAM locations at $2D-$FF which are used by BUFFALO monitor.

The 31K memory locations from $8400 to $FFFF are available as user program CODE or DATA. In assembly language, you specify the starting address with an ORG statement.

You can start DATA RAM at address $00 with the statement ORG 0 followed by RAM variables, as shown below:

```
            ORG   0
TEMP:       RMB   1          ; reserve one byte of RAM for temp storage
XTEMP:      RMB   2          ; reserve two bytes of RAM for temp storage
```

If your program is small, say less than 4K, you can start your program at address $F000 with the statement ORG $F000 followed by your program, as shown below:

```
            ORG   $F000
            LDS   #$FF       ; Initialize the stack point
```

It will assemble your source program and generate hex code within 4K locations from $F000 to $FFFF. If your program is larger, you can change the ORG $F000 to ORG $E000 or ORG $C000. You can use ORG $E000 or ORG $F000 because there is no BUFFALO monitor in the highest 8K locations ($E000-$FFFF)

After fully debugged your code, you do not have to relocate your code and re-assemble it. You can directly program the s19 file into a 68HC711E9. Your code is final for stand-alone operation after finishing your debugging session. It works just like a real time In-Circuit Emulator.

You can abort your program if it's hung in a loop. When you press the ESC key at the PC keyboard, it will interrupt the Husky11 board, but you cannot abort your program with BUFFALO monitor unless you reset the board. The problem with resetting the board is that you would not know where the 68HC11 was hung. It also would leave SWI instructions on all breakpoint addresses resulting in having to download your s19 file all over again.

Here is a very simple program, but it's complete.. It will flash the PB7 LED when it's running.

For a good programming practice, you should always place the LDS instruction in the first line of your code.

```
PB7:        EQU   $80            ; bit 7 of port B
FLS_RATE:   EQU   $8B00          ; change this number will change LED flash rate
            ORG   $F000
START:      LDS   #$FF           ; the top of A1 internal RAM
            LDX   #$1000         ; X register points to the register block
BACK:       BSET  4,X PB7        ; turn on the PB7 LED by setting PB7=1
            JSR   DELAY
            BCLR  4,X PB7        ; turn off the reset LED by resetting PB7=0
            JSR   DELAY
            JMP   BACK
DELAY:      LDY   #FLS_RATE
DLY:        DEY
            BNE   DLY
            RTS
            ORG   $FFFE
            FDB   START          ; reset vector
            END
```

2. Save your file frequently while editing. If you are creating a new file and giving the file a name to save, enter the file name including the file extension, such as “test.asm”, not just “test”.

NOTE: Since the debugger is written in DOS, **folder name and filename** should not be longer than 8 characters.

3. Click Build -> Assemble or click the assembler button on the toolbar to assemble your code. If your code has no errors, it will generate an s19 file, a listing file and a symbol file. If the assembler detects an error, the error message will show the line numbers of your source code that caused error. Beware that sometime (not very often, but it does happen) the freeware assembler may indicate a wrong error line in the file and the actual line that caused error may be one line off.

4. Go to the lines with errors and correct them, then go back to the step 3, until there are no errors.

5. After your code is successfully assembled, you can click Build -> Wytec hc11 tools-> “select this file” button to update the file c:\Ep2IDE\Ep2IDE.txt for the debugger.

6. Close the dialog window and activate the **WBUG11 from the task bar**. Press the F10 function key and the R option. The debugger will automatically download 3 files, namely YOUR_FILENAME.S19, YOUR_FILENAME.SYM, and YOUR_FILENAME.PAR. The S19 file is the hex code. The SYM file is the symbol file, so you can use symbols in commands instead of hex numbers. The PAR file is the parameter file that includes the EEPROM programming enable/disable flag, INIT, TMSK2, OPTION, and BPROT register values. The PAR file also includes breakpoint addresses and the memory display addresses.

The assembler makes a default PAR file. When you exit the debugger, the debugger will update it by saving the current settings. **The PAR file is automatically saved after you exit a debugging session.**

Also, the Write command of the debugger can create a PAR file. After invoking the Write command in the debugger, it will prompt you to enter a choice among U, S, and P options. Choose the P option and give a full file name such as test.par. It will make a PAR file for you.

The most useful feature of the PAR file is its ability to remember the TMSK2 register value. Assuming you changed pre scale bits in the TMSK2 in the beginning of your source program (the TMSK2 register can only be changed in the first 64 E cycles in expanded mode), you have to use the Init command to change the TMSK2 register to match that value before running your program under the WBUG11, otherwise your program will not run. When the PAR file is loaded, it will generate a system reset and force the 68HC11 to take the new TMSK2 register value from the PAR file. For more information, see the Questions & Answers section of the Wytec user manual.

7. When the download is finished and you are prompted with Ep2711>. Next, the PC will reset the board. During the reset, there will be some communication between the PC and the board as shown by the UTX and URX LEDs. If you enter the command too soon, it could disrupt the communication and you will get an error message. So you should wait until those two LED stop flickering After this reset process, you can run your program by entering “go \start” where the start is the label of the starting address in your source code or enter G F000 if you know the starting address is $F000, but do not enter command too soon

In the command line, **ALL NUMBERS ARE HEXADECIMAL** and the $ sign is not needed. Also you should notice that the label \start has the back slash in the front. **All hex numbers may be substituted by symbols starting with the backslash ‘\’.** For more information read the Command format section in the Wytec user manual.

8. During a debugging session, if you want to modify your source code, you don’t have to exit the debugger. You can activate AsmIDE to edit your code and save your new code with the same file name. Then click Build -> Assemble or click the assemble button on the toolbar to assemble the code. A new S19 file should be generated. Because you did not change the file name, you don’t have to **click the “Select this file”** button again. The only thing you have to do is to click the “**WBUG11**” button at the task bar at the **bottom line of the debugger screen** to activate debugger window. Then enter the F10 key and the R option to download your files and start debugging again.

Single Board Computer application:

The Memory Map in Single Board Computer application:

$0000-$00FF	On-chip 256 byte RAM for the 68HC11A1
$0000-$01FF	On-chip 512 byte RAM for the 68HC11E1
$0800-$0FFF	Wytec monitor in U2, some subroutines are callable from user programs
$1800-$1FFF	Wytec monitor in U2, some subroutines are callable from user programs
$1000-$103F	On-chip 64 control registers, including port B ($1004)and port C ($1003)
$1003	Port C for DIPswitch
$1004	Port B for LEDs
$2200-$23FF	65C22
$2201	Port F, LCD port
$2203	DDRF
$2200	Port G, Keypad port
$2202	DDRG
$2400-$25FF	68B50
$2400	UCTRL, USTAT
$2401	UART
$2800-$2FFF	External device, /CS on the pin40 of J1
$8000-$AFFF	U4, RAM for user data
$B000-$FFFF	User code in U3 (20K usable EEPROM)

The important Jumper settings for SBC:

J9	Set for RUN (top position)
J15	Set for U3 (bottom position)

Program external EEPROM in U3 and 512 bytes of internal EEPROM:

You can program EEPROM chip U3 and 512-bytes of internal EEPROM ($B600-B7FF) under control of the Wytec debugger WBUG11. Use the Load command as described on page 10 of the Wytec user manual. The instructions will be shown on the screen when Load command is entered.

If you want to use the board as an SBC, you can program your application s19 file into EEPROM U3. After programming the chip, move the jumper to the bottom position of J15. It will completely bypass both the Wytec's monitor and the BUFFALO monitor. The address range of U3 is changed from $3000-$7FFF to $B000-$FFFF and the address range of U4 (32K RAM) is changed from $8000-$FFFF to $8000-$AFFF. The program code will auto start from U3 after reset or power up.

You can only program U3 under the control of Wytec debugger WBUG11 and your application s19 file must be assembled in the address range of $B000-$FFFF. **The two highest addresses $FFFE and $FFFF in your program must be loaded with the starting address of the program**. Here are the steps to program U3 with the file test.s19:

Before programming, the jumper on J15 must be placed in the "WYTEC" position and the jumper on J9 must be placed in the "RUN" position (not the "PRG" position).

1. Move the jumper on J21 to the right position to enable EEPROM write.
2. Enter Load command and select the option E to download an s19 file into U3.

3. Enter test.s19 as the file name to be downloaded.
4. When the programming cycle starts, the PB0-PB7 LEDs will march from left to right.
5. The programming is done when the LEDs stop marching.
6. Move the jumper on J21 to the left side to write-protect U3.
7. Move the jumper on J15 to the bottom position (labeled with the 'U3').
8. Press the reset button, your application program should run.

Before programming you have to make sure that the jumper on J21 is in the right side to enable EEPROM write, otherwise the programming process will not continue. After programming, move the jumper to the left side of J21 to write-protect the EEPROM, otherwise the EEPROM could lose data quite easily after running a bad program during a debugging session.

The procedures to program the 512-byte internal EEPROM ($B600-B7FF) are the same as the above steps except for step 2. You need to select the option B instead of the option E after the Load command is entered.

The J21 is used to write-protect the external EEPROM U4 and it has nothing to do with the 512 bytes of internal EEPROM.

About AsmIDE:

AsmIDE is written by Eric Engler and it's simple and easy to use. It offers all the basic features that you need to learn 68HC11 programming with this board. There are quite a few free IDE available, but only AsmIDE and MiniIDE (www.mgtek.com/miniide) are two popular IDEs that include a 68HC11 assembler.

One valuable asset of the Husky11 board is the Wytec debugger, **WBUG11**. If you plan to use BUFFALO monitor, you can use either AsmIDE or the MiniIDE if you are familiar with it. If you prefer to use the Wytec debugger, you need to launch it from the AsmIDE.

Using your own assembler or editor for WBUG11:

If you would prefer to use your own assembler or editor such as the MiniIDE, instead of the AsmIDE, you can use AsmIDE for launching the WBUG11. The full path name of the file that you work on must agree with the file name shown on the Wytec hc11 tools dialog window, otherwise the WBUG11 will not be able to locate your s19 file to be downloaded.

Using the board as a 68HC711E9 programmer:

If you need to program a 68HC711E9, you can use this board as a 68HC711E9 programmer, but only under control of the Wytec debugger WBUG11. You must provide a regulated 12 V DC, 30mA and also make sure that the program is fully debugged. To activate the programmer, click Build -> Wytec hc11 tools, or click the debugger button on the toolbar that will bring up a small dialog window. If the current downloaded s19 file is correctly shown in the dialog window, click the "**Select this file**" button and then click the "**programmer"** button to program the 68HC711E9. The programming is done in bootstrap mode and the programming instructions will be displayed on the screen step by step. The memory addresses range for the 68HC711E9 is from $D000 to $FFFF. If your S19 file contains addresses outside of this range, an error message will be generated and the chip will not be programmed. If your assembler or C compiler generates an s19 file with some extra RAM addresses, **you must use an editor to delete those RAM addresses.**

During the programming, the data will be automatically verified.

The 52-pin PLCC socket is ZIF socket. It will not last much longer than 100 insertions. When the contacts of the PLCC socket are worn, you can use a fine dental tool to pry up all contacts, especially at corners, a little bit to extend its life considerably.

ON-BOARD HARDWARE

In expanded mode, Port B and port C of the 68HC11 are used for address and data buses. They are not available to user programs as I/O ports, but they are emulated by the PRU chip, 68HC24. The PRU stands for port replacement unit.

The emulated port B is an output port and each port B pin is monitored by a LED indicator. The emulated port C is a bi-directional I/O port and it is connected to an 8-position DIPswitch. The DIPswitch is connected to GND via eight 4.7K resistors, so it's not dead short to GND. When port C is driven by external circuits, the DIPswitch setting is ignored, but it's better leave all 8 DIP switches in open positions when the port C is used as an outport port.

The PA0 switch is used as a general purpose input switch, except during power up. During the initial power up, pressing the RESET button momentarily while holding the PA0 switch will force the 68HC11 to enter test mode. In test mode, the configure register can be modified. The PA3 and PA4 headers are the outputs of Output Comparators 3 and 4. They can be used to drive robot servos (Make sure that robot servos have their own power supply).

Port E is an 8-bit ADC or a general purpose input port. The trimmer VR1 is connected to the PE7 input of the ADC port via the J10. The temperature sensor U21 is connected to the PE5 via J37 and the light sensor Q3 is connected to the PE4 via J36.

An on-board logic probe LED is connected to pin 55 of the female socket connector P1F and can be used to monitor high or low state at any point of the circuit as a logic probe. It can be connected to VCC for the power indicator if you add a jumper between pin 55 and pin 57.

U18, 74HC14, generates 38.4K baud for U5, 68B50, and it also provides 38KHz square wave to IR transmitter.

U9, SN75176, converts the TTL signal from the SCI to RS485 differential signals and visa versa.
The terminal block T1 used to daisy chain many Husky11 boards together for a network application. .

Two I/O ports, port F and port G are added through U1, VIA 65C22. These are bi-directional ports. Port F and port G are port A and port B of the 65C22, respectively. The address locations for all ports are as follows:

```
PORTF   $2201
DDRF    $2203     ; 1=OUTPUT, 0=INPUT
PORTG   $2200
DDRG    $2202     ; 1=OUTPUT, 0=INPUT
```

CA2 output from the 65C22 is used to drive the relay K1.
RTS from the 68B50 is used to control direction of RS485 communication. If RTS=0, RS485 port, U9 DS75176, is set for receiver port; If RTS=1, RS485 port, U9 DS75176, is set for transmitter port.

```
RTS=0   RS485 receiver port
RTS=1   RS485 transmitter port
```

For users' convenience, there are two subroutines added to control RS485. User can call RS485_RECV at $0800 to reset RTS to 0, or RS485_XMIT at $0803 to set RTS to 1.

The following are all I/O subroutines in Wytec monitor that are callable from a user's application program:

```
                ORG     $0800   ; Husky11 Rev. C board I/O routines

RS485_RECV:     RMB     3       ; enables RS485 receiving mode
RS485_XMIT:     RMB     3       ; enables RS485 transmitting mode
GET_DATE:       RMB     3       ; gets current date from PC
GET_TIME:       RMB     3       ; gets current time from PC
OUTSTRG00:      RMB     3       ; outputs a string terminated by 0
LCD_INI:        RMB     3       ; initializes the 16x2 LCD module
LCD_LINE1:      RMB     3       ; displays 16 char on the first line
LCD_LINE2:      RMB     3       ; displays 16 char on the second line
SEL_INST:       RMB     3       ; selects instruction before writing the LCD module
SEL_DATA:       RMB     3       ; selects data before writing the LCD module
WRT_PULSE:      RMB     3       ; generates a write pulse to the LCD module
```

The circuit is designed in such way that the value of all resistors and capacitors are not critical, they can be off -50% or +100%.

How to use port F (LCD port):

The port F is an 8-bit bi-directional port. Its primary usage is for an LCD display module. If the port is not used as an LCD display, it can be used as a general-purpose I/O port that can be accessed via J2 or J3.

The pinouts of the J3 is as follows:

Pin 1	GND	
Pin 2	VCC (5V)	
Pin 3	Via a 100 Ohm resistor to GND	
Pin 4	PF0	RS pin for LCD module
Pin 5	GND	
Pin 6	PF1	EN pin for LCD module
Pin 7	Not used	
Pin 8	Not used	
Pin 9	PF2	
Pin 10	PF3	
Pin 11	PF4	DB4 pin for LCD module
Pin 12	PF5	DB5 pin for LCD module
Pin 13	PF6	DB6 pin for LCD module
Pin 14	PF7	DB7 pin for LCD module
Pin 15	Via an 18 Ohm resistor to VCC	LED backlight for LCD module
Pin 16	GND	

How to use the keypad interface (port G):

Port G is an 8-bit bi-directional port. Its primary usage is for a 4X4 keypad interface. If the port is not used as a keypad interface, it can be used as a general-purpose I/O port that can be accessed via J2 or J6.

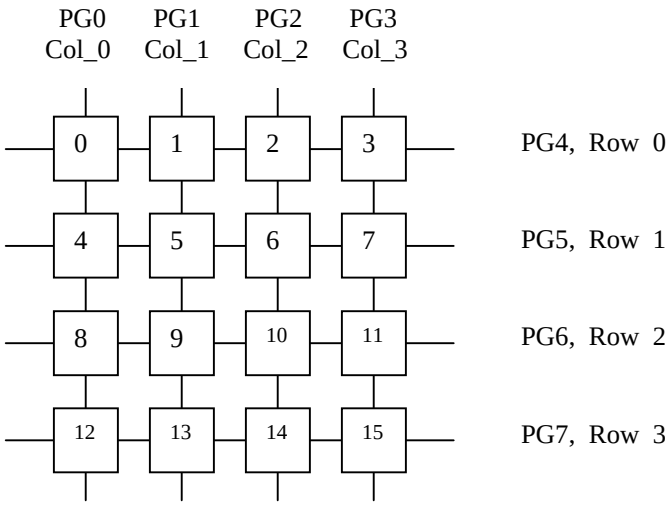
The pinouts of the J2 is as follows:

Pin 1	PF7	Pin 2	PG7
Pin 3	PF6	Pin 4	PG6
Pin 5	PF5	Pin 6	PG5
Pin 7	PF4	Pin 8	PG4
Pin 9	PF3	Pin 10	PG3
Pin 11	PF2	Pin 12	PG2
Pin 13	PF1	Pin 14	PG1
Pin 15	PF0	Pin 16	PG0
Pin 17	VCC	Pin 18	VCC
Pin 19	GND	Pin 20	GND

The pinouts of the J6 is as follows:

Pin 1	PG0
Pin 2	PG1
Pin 3	PG2
Pin 4	PG3
Pin 5	PG4
Pin 6	PG5
Pin 7	PG6
Pin 8	PG7

The following signal definitions only apply to Wytec 4X4 membrane keypad.
Wytec 4X4 membrane keypad connections:



PG0 connects COL0 of the keypad via pin 1 of the 8-pin keypad header J6
PG1 connects COL1 of the keypad via pin 2 of the 8-pin keypad header J6
PG2 connects COL2 of the keypad via pin 3 of the 8-pin keypad header J6
PG3 connects COL3 of the keypad via pin 4 of the 8-pin keypad header J6

PG4 connects ROW0 of the keypad via pin 5 of the 8-pin keypad header J6
PG5 connects ROW1 of the keypad via pin 6 of the 8-pin keypad header J6
PG6 connects ROW2 of the keypad via pin 7 of the 8-pin keypad header J6
PG7 connects ROW3 of the keypad via pin 8 of the 8-pin keypad header J6

The PG0-PG7 has a 100K pull-up resistor in each line.

The keypad scan routine sets PG3 low, PG0,PG1,PG2 high, and then test the PG4-PG7
If no key is down, PG4-PG7 remain high.
If PG7 = low, the key 15 is down.
If PG6 = low, the key 11 is down.
If PG5 = low, the key 17 is down.
If PG4 = low, the key 3 is down.

The keypad scan routine then sets PG2 low, PG0,PG1,PG3 high then test the PG4-PG7
If no key is down, PG4-PG7 remain high.
If PG7 = low, the key 14 is down.
If PG6 = low, the key 10 is down.
If PG5 = low, the key 6 is down.
If PG4 = low, the key 2 is down.

The keypad scan routine then sets PG1 low, PG0,PG2,PG3 high then test the PG4-PG7
If no key is down, PG4-PG7 remain high.
If PG7 = low, the key 13 is down.
If PG6 = low, the key 9 is down.
If PG5 = low, the key 5 is down.
If PG4 = low, the key 1 is down.

The keypad scan routine then sets PG0 low, PG1,PG2,PG3 high then test the PG4-PG7
If no key is down, PG4-PG7 remain high.
If PG7 = low, the key 12 is down.
If PG6 = low, the key 8 is down.
If PG5 = low, the key 4 is down.
If PG4 = low, the key 0 is down.

SPI port pinouts are as follows:

Pin 1	VCC (5V)	Pin 2	VCC (5V)
Pin 3	PF2 (LOAD)	Pin 4	PD2 (SPI DATA IN)
Pin 5	PF3 (STROBE)	Pin 6	PD3 (SPI DATA OUT from 68HC11)
Pin 7	not used	Pin 8	PD4 (CLOCK)
Pin 9	GND	Pin 10	GND

All on-board jumpers:

J1 40 pin female connector for address and data ports, Freescale 68HC11 EVM compatible
J3 LCD port for an LCD module, 4-bit data interface
J4 SPI connector
J5 Mode selector. MODEA and MODEB jumpers are not used in debugging sessions. They can be used for programming the 68HC711E9 OTP part in bootstrap mode.

J6 4 X 4 keypad interface
J8 The 68HC11's SCI receiver source selector (numbering from top to bottom)
1= SCI PD0 receives signal from USB PORT if the two jumpers on the J32 are in bottom position
2= SCI PD0 receives signal from J33, the TTL logical level of RS232 input from your target board
3= SCI PD0 receives signal from T1 (Terminal Block) for RS485 input
4= SCI PD0 receives signal from the on-board IR receiver

J9 The RUN/PRG jumper is used for selecting an operating mode
When it's in the "top" position, the board is used for debugging your code.
When it's in the "bottom" position, the board is used **ONLY** for programming the 68HC711E9 chip

J10 Connects VR1 trimmer pot to PE7 of ADC.
J11 Analog voltage reference. It's connected to the on-board 5V DC reference voltage via a cut-off trace.

J13 Clock output. It's not installed. If it's installed, the clock output of the 68HC11(pin 8) is connected to the pin 8 of the P1 and it can be a clock source of a user target board.
J14 Enables speaker.
When it's in the "top" position the speaker is driven by PA5, Output Comparator 3
When it's in the "bottom" position the speaker is driven by OUTB of the DAC, LTC1661
J15 Monitor selector. Place a jumper in the "top" position for Wytec monitor, the "middle" position for BUFFALO monitor and the "bottom" position for auto-starting program in U3.

J17 IR transceiver control source selector.

When jumpers are placed in the "up" position (labeled with 'SCI'), the 68HC11's PD1drives IR transmitter and 68HC11's PD0 receives data from IR receiver. The PD0 and PD1 can be general purpose I/O pins or SCI.

.

When jumpers are placed in the "low" position (labeled with 'PA26'), the 68HC11's PA6 drives IR transmitter and PA2 receives data from IR receiver.
J18 Enables the 7-segment LED display driver U11, 74HC367.
J19 Enables LCD backlight. The jumper can be removed for extending the life of the backlight.
J20 External 12V programming voltage source.
J21 U3 EEPROM write protect, place a jumper in the "left" position to disable (write-protect) EEPROM programming. Place a jumper in the "right" position to enable EEPROM programming.
J22 Enables BUFFALO trace function.
When jumper is in the "left" position, PA3 is connected to XIRQ to enable single-stepping operation in BUFFALO monitor mode. When jumper is in the "right" position, PA3 is disconnected from the XIRQ.

J26 Servo motor power select. The jumper is placed in the "top" position if servos are powered by the on-board VCC (5V). The jumper is placed in the "bottom" position if servos are powered by an external 5V power supply at the terminal block T4.
J27 Enables RGB LED
J28 VCC for H-bridge driver, U19, SN754410N. The H-bridge driver and 7-segment LED display should not be enabled at the same time. Move the jumper from J18 to this header only if H-bridge driver is used. When H-bridge driver is not used, move this jumper back to J18

J29 DC motor power select. The jumper is placed in the "top" position if motors are powered by the on-board unregulated 9V (VIN). The jumper is placed in the "bottom" position if motors are powered by external voltage at pin 1 of the terminal block T3.

J30 TTL logic level of the ACIA (68B50). In order to use this port, the jumper on the J31 must be set in the "top" position.

J31 The jumper is installed in the "bottom" for USB interface, or in the "top" for TTL interface

J32 Both jumpers are installed in the "top" positions for connecting ACIA (68B50) to USB port.
Both jumpers are installed in the "bottom" positions for connecting SCI (68HC11) to USB port.

J33 TTL logic level of the SCI for user application. In order to use this port, both jumpers on the J32 are installed in the "bottom" positions and the jumper on the J8 must be set in the second position from the "top" labeled with '232'.

J34 X-Y-X Accelerometer module interface or IR distance sensor, GP2D12, interface.

J35 Connects CA2 of the U1 (6522) to relay circuit.

J36 Connects light sensor Q3 to PE4 of ADC.

J37 Connects temperature sensor U21 to PE5 of ADC.

PA3 PA3, OC5 output for servo applications

PA4 PA4, OC4 output for servo applications

PA6 PA6, OC2 output for servo applications

PA7 PA4, OC1 output for servo applications

P1 60 pin male connector for I/O ports, EVB/EVBU compatible.

P1F 60 pin female connector for I/O ports, EVB/EVBU compatible.

P2 USB port for development work, connects to a PC USB port

P4 40 pin female connector for address and data ports, Freescale 68HC11 EVM compatible

MODEA and MODEB jumpers: They are not used in debugging sessions. They can be used for programming the 68HC711E9 OTP part in bootstrap mode or debugging in single chip mode.

BUFFALO I/O routines.

Many 68HC11 books have example programs that access BUFFALO I/O routines. The BUFFALO I/O routines are located at $FFA0-$FFCF. When using it with the Wytec debugger WBUG11, the BUFFALO I/O functions are duplicated at $0FA0-$0FCF.

Following are the BUFFALO I/O routines' function description and jumper table:

```
                ORG     $0FA0

UPCASE          convert the character in A to uppercase
WCHEK           test the character in A for white space and returns with the Z bit set if A is a
                white space  (Space, comma, tab)
DCHEK           test the character in A for white space and returns with the Z bit set if A is a
                carriage return or white space  (Space, comma, tab)
INIT            initialize SCI, is not needed with the Husky11 board
INPUT           reads PC keyboard input
OUTPUT          writes the character in the A to CRT display
OUTLHLF         converts 4 most Significant Bit of A to ASCII and Writes it to CRT display
OUTRHLF         converts 4 most Significant Bit of A to ASCII and Writes it to CRT display
OUTA            output ASCII character in A to CRT display
OUT1BYT         converts the binary byte that is pointed to by X register to 2 ASCII bytes and
                write them to CRT display
OUT1BSP         it's OUT1BYT followed by sending a space to CRT display
OUT2BSP         converts the binary word (2 bytes) that is pointed to by X register to 4 ASCII
                bytes and write them followed a space to CRT display
OUTCRLF         write carriage return, line feed to CRT display.
OUTSTRG         it's OUTCRLF followed by writing the ASCII string that is pointed to by
                X register to CRT display and until character is $04
OUTSTRG0        it's OUTSTRG without writing leading carriage return & line feed to CRT display
INCHAR          waits for an ASCII character from keyboard and put it in accumulator A
```

Jump Table

```
$0FA0           UPCASE
$0FA3           WCHEK
$0FA6           DCHEK
$0FA9           INIT
$0FAC           INPUT
$0FAF           OUTPUT
$0FB2           OUTLHLF
$0FB5           OUTRHLF
$0FB8           OUTA
$0FBB           OUT1BYT
$0FBE           OUT1BSP
$0FC1           OUT2BSP
$0FC4           OUTCRLF
$0FC7           OUTSTRG
$0FCA           OUTSTRG0
$0FCD           INCHAR
```

We have added several I/O routines in the beginning of our Wytec monitor.
Following are the Wytec monitor I/O routines' function description and jumper table starting at $0800:

```
                ORG     $0800
```

RS485_RECV	sets RS485 port to receiver mode.
RS485_XMIT	sets RS485 port to transmitter mode
GET_DATE	X register points to a 10 byte RAM block before calling this subroutine, it returns date information of host PC in the format of MM-DD-YYYY.
GET_TIME	X register points to an 11 byte RAM block before calling this subroutine, it returns time information of host PC in the format of HH:MM:SS AM or HH:MM:SS PM.
OUTSTRG00	It's OUTSTRG0 except the ending character is $00, instead of $04.
LCD_INI	initialize a 16x2 LCD display module
LCD_LINE1	displays 16 characters on the first line of a 16X2 LCD display module
LCD_LINE2	displays 16 characters on the second line of a 16X2 LCD display module
SEL_INST	selects instruction before writing a LCD module
SEL_DATA	selects data before writing a LCD module
WRT_PULSE	generates a write pulse for LCD module

Jump Table

$0800	RS485_RECV
$0803	RS485_XMIT
$0806	GET_DATE
$0809	GET_TIME
$080C	OUTSTRG00
$080F	LCD_INI
$0812	LCD_LINE1
$0815	LCD_LINE2
$0818	SEL_INST
$081B	SEL_DATA
$081E	WRT_PULSE

IMPORTANT NOTES

The following are some important notes that you should know and they may save your time:

1. Things to do if the board does not work.

Many little mistakes can cause a big problem, especially for new beginners. For instance, you may debug your code in expanded mode, but MODEB and MODEA are set for bootstrap mode. If the jumper on J19 is missing, LCD backlight won't work and if the jumper on J18 is missing, 7-segment display won't be lit.

Before troubleshooting the board, you must apply power to the board. When pressing the reset button, the LED should flash twice. If it does not flash, the board may not have 5V DC. You can use a DMM to check voltage at the VCC test point. Sometimes it may be caused by a bad AC adapter or the AC adapter may not even be plugged in.

Sometime the Config register has a wrong value, but the 68HC11 chip is still good.

To determine if the board malfunctions, you can restore the board jumper settings to the original default settings when you receiving the board. The default settings are as follows:

J5 Mode selector. No jumper is installed for expanded mode.
J8 HC11 SCI receiver source selector
The jumper is in “bottom” position (labeled with ‘IR’)

J9 RUN/PRG jumper is used for selecting an operating mode.
The jumper is placed in “up” position for debugging your code.

J15 Monitor selector. The jumper is placed in “middle” position for BUFFALO monitor.
J17 IR transceiver control source selector.
The jumpers are placed in “up” position (labeled with ‘SCI’), the HC11’s PD1 drives IR transmitter and the HC11’s PD0 receives data from IR receiver.
J22 Enables BUFFALO trace function. The jumper is placed in “left” position.

J31 The jumper is installed in the “bottom” position for USB interface
J32 Both jumpers are installed in the “top” positions for connecting ACIA (68B50) to USB port.

If all above settings are correct, when you press the reset button, the RESET LED should flash twice. If it does not occur, you can try to enter test mode to determine if the 68HC11 chip is bad. Sometimes the Configuration register has a wrong value, but the 68HC11 chip is still good and will work in test mode. In test mode it will automatically re-program the value of the Configuration register to $0D. If you cannot enter test mode, the 68HC11 is defective.

To enter test mode, you have to change the jumper on J15 to the “top” position to enable the Wytec monitor. Then press the reset button momentarily while holding down the PA0 switch to enter test mode. The RESET LED should flash twice and the mode indicator ‘EXP’ on the top line of the debugger screen should be changed to ‘TST’.

The crystal is not soldered to the board and it’s held by two machine pins. If you want to change the crystal, just unplug it and replace it with a new frequency, such as 9.8304 MHz. Don’t cut the crystal’s leads too short. If it’s shorter than ¼”, its metal case could short the two machine pins.

2. Always reset the board before downloading a new program.

If a previous application program that you ran was aborted, then you may need to reset the board before downloading a new application program. The reset action will disable the interrupt that was enabled by the previous application. If the interrupt was caused by a timer and is not disabled, the timer interrupt will continue even it’s not called for in your new application program. The result will be unpredictable.

3. Keep folder name and file name not to exceed 8 characters when working under Wytec monitor.

The Wytec debugger is a DOS application, so keep folder and file names short, otherwise the debugger may not be able to find your application program.